ADDING AN INTELLIGENT TUTORING SYSTEM TO AN EXISTING TRAINING SIMULATION

Richard H. Stottler and Randy Jensen
Stottler Henke Associates, Inc.
San Mateo, California

Bill Pike
STRICOM
Orlando, Florida

Rick Bingham
MÄK Technologies
Cambridge, MA 02138

Battle Command 2010 (BC2010) is a tactical decision game used by Command Prep Course students at the Command General Staff College at Fort Leavenworth to play battalion level tactical scenarios in a dynamic, 3-D environment. The use of this simulation, however, still required the effort of an instructor to observe the student's actions and provide an after action review (AAR). It was determined that the addition of an Intelligent Tutoring System (ITS) to BC2010 would off-load the instructor from these duties and allow the students to execute scenarios without requiring an instructor for the AAR. This paper presents the lessons learned from that experience.

In BC2010, students playing a scenario must first read the mission background which includes the mission objectives and five paragraph order. They then develop a plan and input that plan into each unit under their control. They monitor the execution of their plan and the tactical situation in 2-D and 3-D views. Enemy units are only shown when they are sighted by friendly units. During the simulation the students can issue real-time commands.

The ITS is interfaced to BC2010 via the High Level Architecture (HLA.). Initially the student's plans are transmitted from BC2010 through HLA to the ITS, before simulation execution begins. These plans are critiqued by the ITS by comparing them to good and common bad plans for the scenario, as determined by a subject matter expert. The student receives this feedback and corrects the plan. Execution then begins. BC2010, through HLA, sends to the ITS both the locations and actions of vehicles and the commands sent by the student. The ITS evaluates the correctness of these actions, given the current circumstances, determines which tactical principles the student has correctly applied and which have been missed, and automatically assembles a debriefing. It can then recommend further study and additional scenarios to improve the student's weakest areas.

Biographic Sketches:
Dick Stottler co-founded Stottler Henke Associates, Inc. (SHAI), an artificial intelligence consulting firm in San Mateo, California in 1988. He has been principal investigator on a number of tactical decision-making intelligent tutoring system projects conducted by SHAI and is working on an ITS for battalion commanders at the Command General Staff College and a prototype for the future combat system. He has a master's degree in computer science from Stanford University.

Bill Pike is the Lead Principal Investigator for Advanced Distributed Learning (ADL) with the US Army STRICOM. At STRICOM, he has been the principal investigator on several ADL topics on integrating courseware with advanced topics, including the use of game engines as assessment tools for tactical principles. He is an officer in the Naval Reserves and has a Master's degree from the UCF in Computer Engineering.

Mr. Richard Bingham is the Director of Programs at MÄK Technologies where he is responsible for the daily operations of all government contracts relating to MÄK's SIM*inter*NET, PC-based training applications business. Before joining MÄK, Mr. Bingham worked for Sikorsky Aircraft. As a simulation engineer he was responsibly for the physics based modeling of helicopter dynamic components and Mission Equipment Package components.

Randy Jensen is a Project Manager for Stottler Henke Associates, Inc. (SHAI) and has developed intelligent tutoring systems and prototypes for a variety of domains, including current projects for battalion commanders and personnel at the Air Force's Aerospace Operations Center. He has a BS in Symbolic Systems from Stanford University.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **2006**	2. REPORT TYPE	3. DATES COVERED **00-00-2006 to 00-00-2006**
4. TITLE AND SUBTITLE **Adding an Intelligent Tutoring System to an Existing Training Simulation**		5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Stottler Henke Associates Inc,951 Mariner's Island Blvd Suite 300,San Mateo,CA,94404**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES **The original document contains color images.**		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**10**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

ADDING AN INTELLIGENT TUTORING SYSTEM TO AN EXISTING TRAINING SIMULATION

Dick Stottler
Stottler Henke Associates, Inc.
San Mateo, California

Bill Pike
STRICOM
Orlando, Florida

Rick Bingham
MÄK Technologies
Cambridge, MA 02138

Randy Jensen
Stottler Henke Associates, Inc.
San Mateo, California

PROBLEM DESCRIPTION

Instructors and experts agree that the most important factor in developing skilled tactical decision-making is practice making tactical decisions in tactical scenarios. The Department of Defense (DoD) has long recognized this as well. They first addressed it primarily through field exercises. These were expensive and available only a small fraction of the time. This problem was addressed through the development of simulations. But these were originally large, expensive, and costly to operate.

In the last five years, much progress has been made in developing low-cost simulations to support military training and education. Likewise, the military recognizes the potential of advanced distance learning and is striving to maximize its use for education and training. These efforts have, in many cases, reduced the overhead in computers and manpower needed to train our soldiers. Today, soldiers, using a single personal computer, can practice their tactical skills in a simulated battlefield environment. Only a few years ago, this same training would have required multiple computers, several computer operators, observer-controllers, and an exercise coordinator. Battle Command 2010 (BC2010) is a tactical decision making game. It is an example of how a low-cost simulation can be used to enhance advanced distance learning.

BC2010 represents the next generation of tactics trainers. However, one hurdle remains before the power of the PC-based tactical simulation can be harnessed: the incorporation of an intelligent tactical tutoring system. Until this is done, an instructor is needed to monitor the soldier's actions and decisions, recognize the critical teaching points, coach higher levels of performance, and provide remedial instruction.

BC2010 DESCRIPTION

Battle Command 2010 (BC2010) was developed for the Battle Command Battle Lab at Fort Leavenworth, Kansas. Part of the Intermediate Desktop Trainer (IDT) genre of simulations, BC2010 provides an environment for Army Commanders and their staffs to evaluate skills in planning and executing tactical operations from a brigade level and below. The system has been designed to allow both single-player and multi-player operation for collaborative execution and can be used for head-to-head game play. Using standard desktop personal computers, the trainer has an embedded simulation engine that is capable of modeling opposing red forces and adjacent blue forces. Additionally, a built-in after-action-review (AAR) system is available to record the complete networked mission, playback the mission on a 2-D map or 3-D environment, and create statistical charts for analysis.

BC2010, in its simplest form, is played in a single player mode. In this mode, the commander is responsible for planning all battlefield functional areas (BFAs) and during execution, commanding all units. The trainer, however, is more commonly played with several users over a network. In this mode, one player traditionally assumes the role of the brigade commander. The other players, during planning, assume the roles of the staff officers (e.g., Maneuver, Fire Support, Engineering, Intelligence) and during execution, the unit commanders.

Once a mission is selected, both planning and game play are performed through an intuitive game-like user interface, shown in Figure 1. BC2010 has two primary displays, the 3-D "pop the hatch" stealth view and the 2-D map view. The 3-D view is used so the player can view the outside world and obtain first hand knowledge of the outlying terrain. This view provides a realistic representation of the world. The 2-D map view is used for all planning and execution-based activities.

During planning, the map view permits the user to graphically plan a BFA and view the plans from other BFA's through the use of multiple tactical overlays. These overlays, shared over the network in a multi-player game, represent the acetates that would be used by the staff officers in the tactical operations center (TOC). Objects placed on the overlay are used to portray the intentions of how the player will perform that portion of the operation. For instance, a maneuver staff officer might place several routes leading to battle positions and support by fire positions, all graphically represented with associated locations. The player can then use the graphics on the overlays (e.g., a planned minefield) as objects in the simulation on which subordinate units can be assigned to perform tasks. Thus, all simulated units can be assigned plans to be performed during execution.

During execution, the user shifts gears and is quickly engaged in an ever-changing battle. The user utilizes the map view to monitor progress of the troops and make modifications to units' plans as required. Complex, multi-phased plans can be modified as new enemy intelligence is gathered, or instant action commands can be issued to units as time becomes more critical. In multi-player missions, communications during execution are performed through the use of a text-based chat or a voice-over-IP system, permitting collaboration in the tactical decisions to be made.

Figure 1. Battle Command 2010 Game Interface

ITS CONCEPT DESCRIPTION

Intelligent tutoring systems (ITS) can best be defined as advanced training software that mimics a human tutor by adapting its instructional approach to each individual student. They are particularly valuable for teaching complex cognitive tasks such as trouble shooting, problem solving, and resolving critical situations. As a human tutor does, an ITS continually monitors and assesses each student's actions, infers the student's state of knowledge, and decides on the next instructional event to maximize the student's learning. To do this in a significant and cost-effective way, intelligent tutoring systems use artificial intelligence.

One-on-one tutoring by skilled human tutors is widely regarded as the single best mode of instruction. A study by Benjamin S. Bloom of the University of Chicago and Northwestern University concluded that, under the best learning conditions they could devise (tutoring one-on-one), the average student was 2 Sigma above the average control student taught under conventional group methods of instruction. That is, the average tutored student was above 98% of the students in the control class.

Conventional Interactive Multimedia Instruction (IMI) software is not designed to provide such a high level of adaptive response to each individual student as an individual human tutor or ITS can. In fact, most IMI software more closely resembles an "electronic textbook" rather than an "electronic teacher." Just as a book implicitly encourages a student to start at the front and move to the back, such IMI software usually encourages a student to move linearly through a set of multimedia material, with the occasional multichoice questions to test the student's retention of the information. Such tests do not assess the student's ability to apply the information. The ability to apply information in a job should be the goal of training.

Also, conventional IMI is not able to meaningfully incorporate use of free-play simulators into their curriculum. This is a major shortcoming of conventional IMI as student manipulation of sophisticated simulators that realistically replicate issues that they will encounter on the job is widely recognized to be a highly effective training technique. The catch has been that simulators without instructors are virtually useless for training, and their unsupervised use can even result in negative training. Students working on simulators need instructors to point out their correct and incorrect actions, to brief them in context on the underlying concepts that are being taught, and to decide on the next appropriate simulated scenario for the student to run.

Intelligent tutoring systems are ideal for incorporating desktop free-play simulators into computer-based training since the software can stand in for a human tutor in all the roles. Existing IMI course material can often be integrated with ITS-enabled simulator and other active training. In this way, ITS technology can greatly leverage the training value of existing IMI and desktop simulators. As shown in Figure 2, the ITS can monitor a student's interaction with both simulation and other training content in the IMI, create and update the student model and decide on the next instructional event (e.g., provide a hint, ask a question, run a new scenario, display multimedia to explain a concept, alert a human instructor that the student needs special help, etc.).

To keep track of each student, the intelligent tutoring system creates and maintains a "student model" for each individual from the first time he or she logs onto the software. Depending on the sophistication of a particular intelligent tutoring system, the student model keeps different amounts of information on the student. The most basic information includes the tasks the student has performed as well as performance information on those tasks. From this information, the software estimates the student's mastery of relevant skills and knowledge, and the student's ability to apply them when appropriate. For example, a student may be able to apply a concept in one set of circumstances, but not under other circumstances, so it is important that the software tests the student's knowledge of each concept under different circumstances.

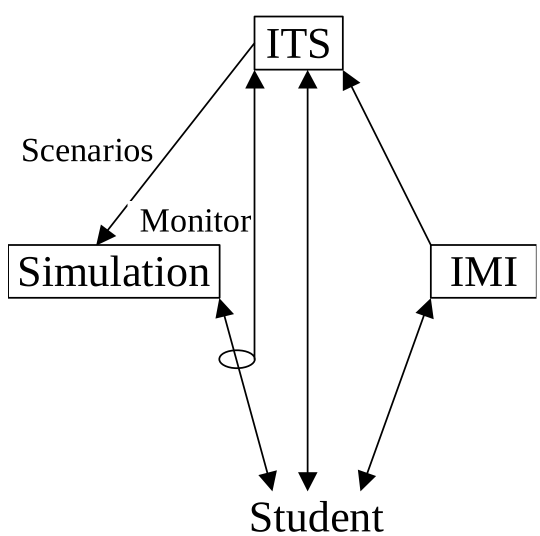

Figure 2. ITS can integrate free-play simulators and IMI

BC2010 ITS DESCRIPTION

Overview

Figure 3 illustrates the interaction between BC2010, ITS, and the student in the ultimate configuration. During the planning phase, the student obtains preliminary plan information from the courseware and develops a detailed plan in BC2010. Once the plan is complete, the ITS evaluates the plan and provides feedback through the BC2010 interface to the student. This is provided to the student through annotated tactical overlays and animations that illustrate the likely outcomes of a bad plan. The student is able (and encouraged) to modify the plan, and then moves to the execution phase. During execution, the student receives both 2-D and 3-D data from the BC2010 interface and provides command and control information to the BC2010 simulation engine. In the background, BC2010 sends the ITS the internal data necessary for analysis. By monitoring the student actions in the simulated scenario, the ITS assesses their correctness in the current situation. The results are used to debrief the student by automatically assembling an ITS-based After Action Review (AAR). The AAR will ultimately be provided to the student through the BC2010 interface, and will use the current BC2010 AAR capability as the baseline. Currently, the ITS uses its own interface for all debriefing. After the student finishes the scenario, the ITS will infer the knowledge deficiencies of the student and formulate a remedial instruction plan, which normally includes further course material, examples, and further BC2010 exercises, based on Command Prep courseware, to practice and test the student's weaknesses.

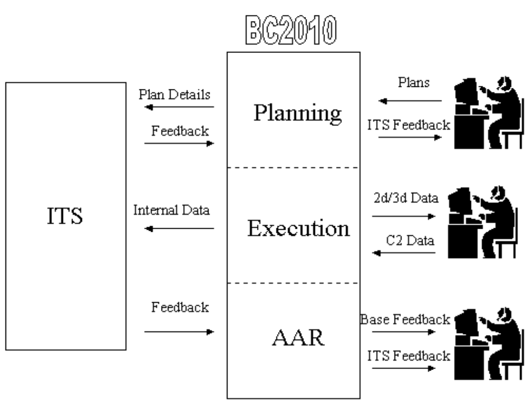

Figure 3. Interaction Between BC2010, Intelligent Tutoring System, and Student

Currently, the ITS acts as a separate application and provides feedback through its own user interface. The second integration effort will integrate the ITS more thoroughly into the BC2010 application, as shown in Figure 3.

The integration of an existing simulation with an existing Intelligent Tutoring System has several advantages.

- **Leverage of familiar tactical training tool:** Both students and instructors at CGSC are familiar with BC2010. In addition, CGSC plans on using BC2010 as their simulation tool during a pilot program in 2002.

- **Leverage of existing intelligent tutoring system:** STRICOM leveraged their investment in the existing ITS technology as opposed to developing an intelligent tutoring system from scratch. The two contractors (the ITS developer and the developer of BC2010) did need to work closely to ensure successful integration of the ITS with BC2010.

Tactical Plan Evaluation

Tactical plan evaluation requires the capability to evaluate two factors: (1) placement of the correct kinds of graphical overlay elements in the correct locations within a predefined tolerance, and (2) suitability of the roles assigned for each unit in coordination with overlay elements. Suppose for a simplified example, that Figure 4 is a correct plan supplied by an expert. The instructor also annotates the correct plan, in a separate file, with an overall description of the concept of operations, the rationale for why that concept is a good solution, and the principles that the student must understand to have arrived at this plan. The annotation files also allow the instructor to create similar annotations for each individual symbol. These are descriptions, rationale, and principles for the five elements of the symbol - why the tactic represented by the symbol is needed, why the type of unit or maneuver was chosen, why the size of unit was chosen, why the general location was chosen (which usually relates to tactical considerations of the overall plan) and why the specific location was chosen (which usually relates to terrain features). All of these descriptions and rationale take the form of referenced multimedia files so that animations, graphics, and other multimedia can be used in the explanations, instead of simple text.

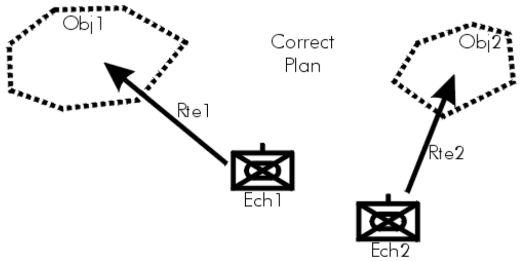

Figure 4. Sample Correct Plan Supplied by Expert

Ech1 and Ech2 represent two companies treated as echelons 1 and 2 for this example. Obj1 and Obj2 are two objective regions and Rte1 and Rte2 are the appropriate routes for echelons 1 and 2 respectively. The student is presented with the same scenario and any background information or intelligence, but without the route arrows. The objective areas may or may not be provided, depending on the nature of the scenario. For example, a trainee may be expected to determine on his own what the effective boundaries should be for the objective areas, given terrain features or other factors. All elements of the correct plan are represented in a hidden layer of the overlay which is only viewable by instructors or course developers. Figure 5 shows an incorrect plan that fails the first evaluation factor in two ways.

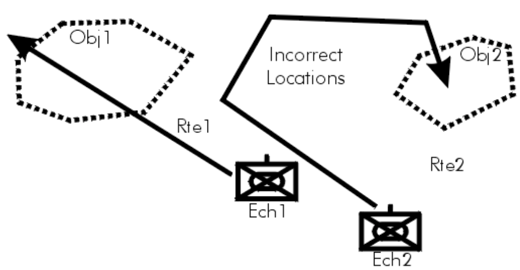

Figure 5. First Incorrect Plan as Entered by Student

Rte1 is incorrect because the end point or destination is outside of the effective area of Obj1. This kind of example may happen in cases where the student is required to determine where the objective area should effectively be, so in this case the student may have misread a terrain feature. Rte2 is incorrect even though it has the correct end point at Obj2, because the route clearly does not match the correct route in the correct plan within any reasonable tolerance. This student would receive in his planning debrief the instructor-entered multimedia rationale for the exact location for Rte1's end point (perhaps that location represents a piece of key terrain) and the rationale for the general and exact location for the correct Rte2.

Figure 6 shows an incorrect plan that fails the second evaluation factor, the assignment of roles.

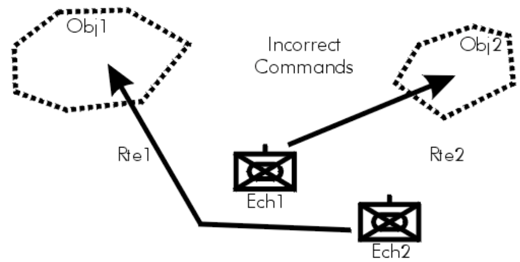

Figure 6. Second Incorrect Plan as Entered by Student

In this case, the student understands the correct routes and objectives, but issues commands incorrectly, in the sense that the wrong units are sent to the wrong objectives, possibly presenting time-space-distance problems and also potential coordination problems as units move across each other.

Tactical Execution Evaluation

The BC2010 environment presents free play simulation of battlefield conditions, so the ITS evaluation of student performance often depends on the observable accomplishment of certain simulation states that involve the relative positions, orientations, and activities of more than one coexisting simulation element. Finite State Machine (FSM) evaluations provide an effective means for monitoring simulation states and triggering analytical conclusions when certain conjunctive conditions are met. Figure 7shows an example of a simple scenario in which student performance is evaluated with respect to a conjoined set of simulation elements.

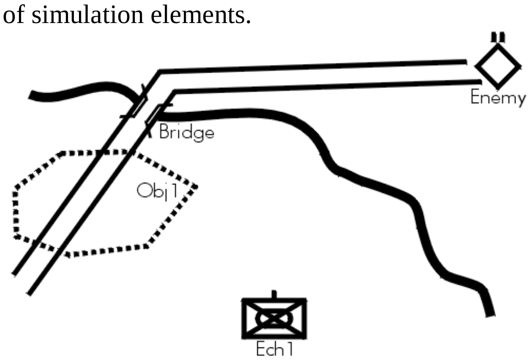

Figure 7. Scenario with Conjoined Simulation Elements

In this scenario, we can imagine that the student is tasked with blocking an approaching enemy unit, given the intelligence report that the enemy battalion is approaching on the road shown. In this case, Obj1 represents an area visible only to instructors, not to the student. The definition of the boundaries for Obj1 becomes necessary for the evaluation of the student's performance in terms of reaching an effective defensive

position. So in this example, the evaluation engine checks for a simple set of conditions, using two functions – GetCurrentPosition, which returns the given entity's position, and PositionMatchesElement, which checks if a given position matches with a given element from the overlay. This is illustrated graphically in Figure 8.

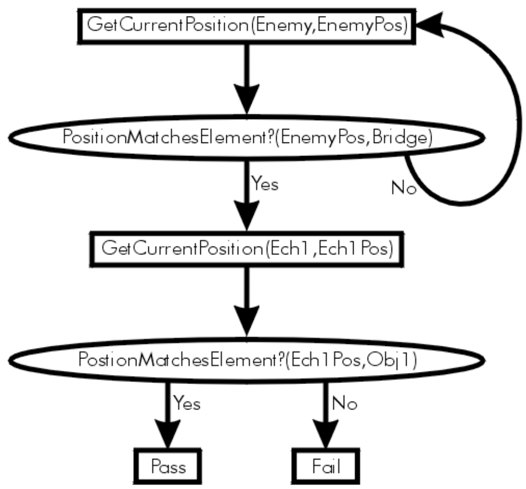

Figure 8. Finite State Machine Example

This simple FSM reaches the "Pass" state if the student has successfully brought the unit designated as echelon 1 to the objective 1 area before Enemy reaches Bridge. This transition would also send a message to the ITS. This message would include a debriefing message that the action was correct and why. This would have been previously attached to the transition by the instructor. These messages become the basis for briefing and real-time coaching as described in the next section.

Remediation

The ITS remediates the deficiencies it finds in several ways. One of the most important is the debrief (also called the after action review, or AAR) which the ITS assembles automatically. There are two types, since the student interacts with the combined system in two phases - pre-mission planning and real-time mission execution.

The pre-mission planning debrief, as described above, is generated by assembling the proper multimedia rationale explanations for the parts of the student's plan that did not match the correct plan. However, this method only is applicable if the student's plan is reasonably close to the instructor's. This usually means that they are in at least rough agreement about the concept of operations. To alleviate this constraint, the instructor can store several correct plans as well as several common incorrect ones. The ITS will first compare the student's plans to all of the plans created by the instructor for the scenario and pick the closest matching one. It then assembles a debriefing based on that one. For common student mistakes, instead of the rationale explaining why the plan's overall concept was chosen, it explains why the overall concept is bad. If the student matches a bad plan, in addition to the explanations as to why it is bad, the student will also get a description of a good one and why it is considered good. The plans and plan symbols also have principles to be passed or failed depending on whether or not the student's symbols match them. In this way, the process that assembles the debriefing (picking the most closely matching plan and comparing its symbols to those of the student's plan) is also used to assemble lists of which principles the student successfully applied in a mission planning context and which ones he could not. This is used in further remediation as described further below.

Since BC2010 sends the ITS plan elements as they are created by the student, it is possible for the ITS to provide instruction during the planning process instead of waiting until the plan is complete. We've developed ITSs that present this instruction in two ways. One is in the form of a Socratic dialog. That is, the ITS asks the student general tactical principle questions particular to the specific scenario and the partially completed plan. These prompt the student to think about tactical principles that appear to be lacking based on the plan so far. The other type of instruction is generally termed coaching. Coaching provides hints to the student while he is developing his plan. The best hints are the ones that provide a minimum of information with little specificity yet get the student to apply the appropriate tactical principles correctly in the planning decisions. This is generally accomplished by providing hints that are very general at first and then increasing their specificity as required to elicit a correct decision. Of course the evaluation system has to be kept informed of the degree of hinting required for a student to make each decision. Ultimately, hinting must be withdrawn as the student's mastery increases so that he does not become dependent on it.

Close integration of an ITS with a tactical simulation provides an especially valuable form of remediation during the planning debriefing. When the student has created a bad plan, that plan can be simulated in faster than real-time so that the student can see the unfortunate results of that plan without having to spend the time to execute it.

This also illustrates the importance of debriefing the plan development before moving on to execution for both instructional and automatic execution evaluation reasons.

Without a dedicated plan debriefing, the student who has a poor plan will merely go on to execute it, spending considerable time running the simulated scenario before finally getting the AAR at the end of the simulation. Only then will the student be informed of the problems with the plan, too long after he had completed it, in opposition to the instructional principle of immediate feedback. Furthermore, the student will have spent a large amount of time with this poor plan, reinforcing his memory of the poor plan. If the scenario happened to go well in spite of the poor plan, which often can happen, the student will have favorable memories of the planning mistakes. This is especially true when compared to the relatively small amount of time the student will spend in the debriefing of his poor plan. By debriefing the poor plan immediately and directing the student toward the development (and then execution) of a good plan, only the positive plan will be reinforced. Finally, it is much easier for the ITS to accurately evaluate the student's performance when the student is executing one of a few known good plans.

The real-time mission execution debriefing messages are assembled as described above by the transitions in the Evaluation FSMs. The transitions also generate lists of passed and failed principles. To create the automatic after action review, the ITS gathers the debriefing messages, organizes them and writes a multimedia AAR file organizing the actions, generally in chronological order. The correct actions are generally indicated in green, and they are accompanied by the explanation as to why they were correct along with the principles that the student must have been able to apply in order to have performed this correct action. For actions deemed incorrect, the action is generally colored red and is accompanied by an explanation as to why the action was incorrect along with the principles that the student was not able to successfully apply. (Failure to take a correct action is a common type of incorrect action.) The ITS also writes out important events that don't necessarily correspond to correct or incorrect action of the students but provide important information as to what the tactical situation was at that time so that the AAR file is easier to follow. The ITS also assembles lists of passed and failed principles.

The same information used to compile the AAR file can also be used to provide a real-time coaching component for the student's real-time mission execution decisions. Coaching during a simulated mission is a matter of instructional philosophy. Some would argue that a coaching component is both unrealistic and disruptive. However the alternative is to both allow the student to make a bad decision (or fail to make a good one) and to delay the feedback until the AAR when the student will be informed of the poor decision. In the case where coaching is deemed appropriate, the best hint is the least specific one that allows the student to make the correct decision and is only presented to a student who would make the wrong decision without it. The latter is handled well by the student model. If the student has a poor history with a principle and application of that principle is necessary to make the current correct decision, it is likely that the student will make a poor decision without a hint. Furthermore, a general hint of the form "Consider " along with name of the principle (such as "The Importance of Key Terrain") can be easily constructed without giving much away. In the event the student still takes an inappropriate action, the very specific hint of "do the correct action because ..." may still be better instructionally than a wrong decision and the delayed feedback of the AAR.

As described above, the process of assembling both types of debriefs also generates, for each scenario, lists of passed and failed principles. This allows the ITS to look at a student's entire history with a principle and decide what level of mastery the student possesses and whether the student needs remediation with reference to this principle. This is generally indicated by poor performance with respect to this principle in multiple scenarios so that this type of remediation occurs outside of the specific scenarios in which the mistakes were made. (Scenario specific remediation was already given in the automatic AAR.) Depending on the type of student and the severity of the problems, the student may be given a description of the principle, a detailed description of the principle, examples of the application of the principle in other scenarios, and hints when faced with this principle in future scenarios. All students having problems with a particular scenario, after remediation, would receive additional exercises that require application of the principle both to prove that they can now apply it in an operational scenario and to force them to practice the areas in which they are the weakest.

BC2010/ITS HLA Interface Description

BC2010 was already HLA compliant before this effort began; likewise, the ITS already had the ability to generate an HLA log file from an HLA-compliant simulation run and analyze it. However, there was a desire to make the interface real-time so that real-time instruction (e.g. coaching) could be performed. Consequently the ITS's HLA logger was converted into an HLA listener. Through the standard HLA Real-Time Platform Reference (RPR) Federation Object Mode (FOM), the ITS immediately had access to information adequate for real-time mission execution evaluation. Most importantly this included vehicle positions, velocities, fire events, hit events, and indirect fire events including their type. However, the ITS also

was tasked with evaluating a student's plan, which is not normally transmitted as part of a standard HLA compliant tactical simulation. This additional information was also transferred to the ITS from BC2010 as described below.

The BC2010 simulation environment provides a set of controls for assembling plan information in a distributed setting. With potentially several users viewing the same scenario, a plan can be collaboratively defined and seen at each user’s station. A plan typically consists of graphical elements defined in an overlay for a given map, coupled with specific commands for specific units, which may either refer to graphical elements from the overlay or function as independent commands. An example of a referential command would be MoveAlong(*route*), where *route* is the ID for a route graphically defined in the overlay. An independent command would be MoveTo(x,y,z). Each unit or echelon may have a separate plan consisting of several commands, either referentially related to the overlay or independent, and possibly including triggers based on test conditions.

In BC2010, the same mechanism that is used to create pre-mission plans is also used to issue orders during real-time mission execution. Thus, once the ITS was adapted to read the BC2010 plans through HLA (as described below), it was also immediately able to see the orders that a student was issuing to the units that he commanded. The real-time mission execution evaluation could also consider a student's orders directly, instead of only being able to examine their effect in the movements and actions of the vehicles.

Since both the graphical overlay elements and the echelon commands are issued in real-time, the BC2010 application has a standard procedure for distributing this information via HLA to all user stations engaged in the scenario. BC2010 uses the RPR-FOM to transmit data via HLA, but for this planning application some extensions to the existing FOM were necessary in order to correctly provide plan information.

The DtDataInteraction class is a general class of the RPR FOM for transmitting data, so in the case of plan information, a DtDataInteraction object is published by the BC2010 application via HLA. The ITS, acting as a federate, includes a listener that parses the DtDataInteraction object to determine if it contains plan information; e.g., a new command for a given echelon or a new graphical element in the overlay. If so, it extracts the appropriate information for plan evaluation purposes.

While the ITS was being interfaced to BC2010 it was also undergoing development unrelated to the ITS. This was both positive and negative. On the one hand, developers were already working on BC2010 and were therefore also available to make changes required for the ITS interface and to answer questions from the ITS team and in general, coordinate the development of the combined system. However it meant that the ITS team was forced to interface to and work with a simulation that was undergoing active development, a moving target so to speak.

Results

The result of the first stage of the integration effort is that BC2010 and the ITS are interfaced through HLA. B2010 and the ITS has a coordinated set of predefined scenarios, so that any scenario that the student is using in BC2010 as part of the combined system is known to the ITS, in the sense of having predefined good and bad plans and predefined evaluation FSMs for it. The ITS successfully receives the plan information from BC2010 and debriefs the student on the plan in its own interface. During mission execution the ITS receives the state of the simulated world and the student's actions and successfully evaluates those using its FSMs. Again this debriefing is given to the student through the ITS's user interface.

Lessons Learned

HLA can be used effectively to interface an ITS and tactical simulation. Furthermore, the RPR FOM can be extended to transmit additional, nonstandard information, such as plan overlays.

The ITS that was interfaced to BC2010 already existed and was interfaced to a variety of products to make a logically complete system. This made the overall system very unwieldy and impractical for training. There were significant ease of use, development, and fielding advantages to interfacing the ITS to a simulation product which represents one self-contained solution with all the needed capabilities in one software package. However, interfacing to a simulation under development was difficult. It would have been optimal to interface the ITS to the simulation after the enhancements were complete, if time allowed. It is important to have the ITS team involved earlier so they can influence the design of the simulation and what information will be available to the ITS from the simulation.

Tactical instructors are comfortable generating scenarios with a limited number of likely good and bad

plans. Given this and the similarity of students' planning mistakes, automatic plan debriefings can be constructed by the ITS. These same scenarios can also be reasonably constrained as to the proper actions expected from the student, depending on the situations that will develop. Therefore, scenario-specific finite state machines can be predefined to evaluate the real-time student decisions. In addition to vehicle motions and events, the actual orders from the student are useful for evaluation of student performance.

A two-step integration plan is being executed. The first step was merely to interface the two applications through HLA. The second stage will be to more tightly integrate the applications so the user only interacts through the combined system through the BC2010 user interface. However the first step by itself is meaningful in the sense that students can still make effective use of the more loosely integrated version.

More tactical decision-making practice is needed to train proficient Army leaders. Experts and instructors agree that there is nothing more important than getting tactical decision-making practice in scenarios. Furthermore this practice must be accompanied by an expert debriefing. In general, debriefing needs to be improved and made available automatically so that students can practice away from the schoolhouse. An ITS integrated with a user-friendly tactical simulation is well-accepted by instructors, because they know the importance of this type of practice.

It is most helpful to evaluate and debrief planning before going on to the execution phase of a mission.

Future Work

The current system has been introduced to the instructors of the Command Prep Course at the Command General Staff College at Fort Leavenworth. The second stage of the ITS integration is beginning which will allow a more thorough integration of the ITS into BC2010 and allow additional types of remediation to support the full set described here. Additionally the dual development teams will need to support the use of the system by instructors and students at the CGSC. The combined system will also be introduced to the Armor Captain's Career Course at Fort Knox. They use similar scenarios in their training.

Although not used in the initial implementation, the ITS that was interfaced to BC2010 has a student modeling capability that has been used in other projects. The student model keeps track of the student's strengths and weaknesses, in terms of which principles have been mastered and which have been problematic. Future enhancements to this system will allow a student to view the student model for himself and have the ITS select scenarios that practice the areas in which the student is the weakest.

A more dramatic extension would extend the combined system to teams of students working in cooperation on a single scenario. BC2010 is designed to allow this type of training. The ITS as it is currently configured could evaluate the overall performance of the team, in the sense of evaluating which decisions were good and bad and could model the knowledge of the team as a whole, if it stayed consistent. However it is not set up to attach different decisions to different students in the same scenario. That particular extension would be straight forward. More difficult would be the analysis of the communication that occurred between team members. In the case where that communication is identical to the orders given to software controlled units, the analysis of the decisions relating to making the communication would not be difficult. In the case where the orders were verbal or free form text, the analysis is more difficult; though given the structure of the environment, a reasonable analysis is possible and should be developed.